Enabling Time-Aware Process Support with the ATAPIS Toolset

Andreas Lanz and Manfred Reichert

Institute of Databases and Information Systems, Ulm University, Germany
{andreas.lanz,manfred.reichert}@uni-ulm.de

Abstract. The proper handling of temporal constraints is crucial for business processes in many application domains. Contemporary process-aware information systems (PAIS), however, lack a sophisticated support of time-aware processes. First of all, at design time it should be possible to specify the temporal constraints of a business process. In turn, this should be accompanied by checking the respective time-aware process schema for inconsistencies that may emerge due to hidden interdependencies among the temporal constraints. The latter is crucial to enable a robust and error-free execution of the time-aware process schema. At run time, corresponding process instances need to be monitored for violations of their temporal constraints. This demo paper presents the ATAPIS Toolset for modeling and enacting time-aware processes. The toolset is based on AristaFlow BPM Suite—an industrial-strength process management system. The ATAPIS Toolset enables process engineers to correctly specify and implement time-aware processes. Further, time-aware process instances can be efficiently executed, whilst monitoring their temporal constraints. Altogether, the ATAPIS Toolset covers the temporal perspective of processes at design as well as run time in a comprehensive way.

1 Introduction

Time is a crucial factor regarding business process support [8]. In many application domains (e.g., patient treatment, automotive engineering) the proper handling of *temporal constraints* is vital in order to successfully execute and complete processes [2,5,8]. Contemporary process-aware information systems (PAIS), however, lack a sophisticated support of *time-aware processes* [8]. To remedy this drawback, the proper integration of temporal constraints with both the design and run-time components of a PAIS has been identified as a key challenge in the development of next generation process management technology [2,5,7].

The toolset presented in this demonstration has been developed in the ATAPIS[1] project. The major goal of this project is to provide a comprehensive framework enabling the specification, enactment and monitoring of time-aware processes in adaptive PAIS. The ATAPIS Toolset[2] allows specifying time-aware

[1] **A**daptive **T**ime- **A**nd **P**rocess-aware **I**nformation **S**ystem
[2] A screencast is available on `dbis.info/atapis`

process schemas as well as checking their temporal consistency at design time. Furthermore, at run time, related time-aware process instances may be created, executed, and continuously checked for temporal constraint violations.

The components presented in this tool demonstration allow modeling temporal constraints as first class citizens of a process schema. In particular, the toolset covers most of the time patterns introduced in [8]. Empirical evidence from case studies has confirmed that these patterns are required for modeling the temporal perspective of processes in a variety of domains. Furthermore, the demo shows how the ATAPIS Toolset enables soundness and consistency checks of time-aware process schemas to guarantee for a robust and correct process instance execution. Finally, different notions of temporal consistency (e.g., weak / dynamic consistency, controllability) are supported. Note that a discussion on further related work is omitted for lack of space. Interested readers are refereed to [8,7].

2 The ATAPIS Toolset

The ATAPIS Toolset is implemented based on the *AristaFlow BPM Suite*, an industrial-strength process management system that exploits advanced process support features we developed in previous research projects [4]. In particular, AristaFlow provides an open API in combination with a modular, service-oriented system design [6]. The latter allows us to extend AristaFlow with functions required for the specification and execution of time-aware processes.

2.1 Specifying Time-Aware Process Schemas

Fig. 1 depicts the process editor of the ATAPIS Toolset. It is based on the *AristaFlow Process Template Editor*, which we enhance with capabilities and language elements required to capture and implement fundamental time patterns [8]. Currently, the ATAPIS Toolset covers the following time patterns: *Time Lags between two activities* (TP1), *Durations* (TP2), *Fixed Date Elements* (TP4), *Schedule Restricted Elements* (TP5), *Time-Dependent Variability* (TP8), and *Cyclic Elements* (TP9).

In order to enable *time lags between two activities* (TP1) and *cyclic elements* (TP9) we extend AristaFlow's process modeling language [10] with *time edges*. In the process editor, such a time edge between two activities is visualized by a dashed line (cf. Fig. 1). In accordance with TP1, a time edge may be added between arbitrary activities that can be conjointly executed in the context of a process instance (i.e., a time edge must not be added between activities from exclusive branches). Furthermore, a time edge is configured in respect to the represented restriction (i.e., minimum and/or maximum time distance) and the kind of time lag (i.e., start-start, start-end, end-start, or end-end). The concrete configuration settings are reflected by a label attached to the time edge; e.g., the label $E[0h, 2h]S$ which is attached to the time edge between activities `prepare patient` and `perform treatment` (cf. Fig. 1), describes a time lag with a minimum time distance of $0\,hours$ and a maximum time distance of $2\,hours$ between the end (E) of the former activity and the start (S) of the latter.

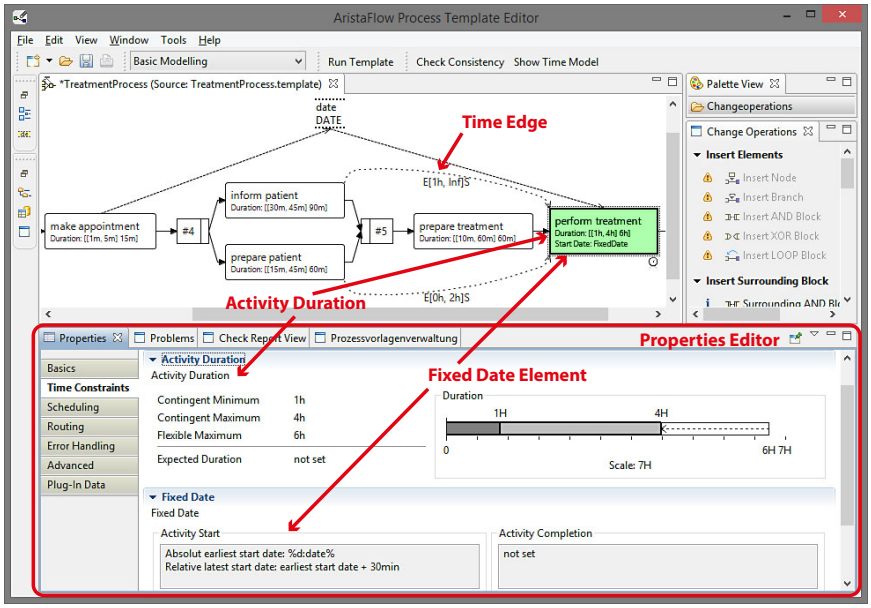

Fig. 1. ATAPIS process editor enabling the modeling of temporal constraints

In turn, time patterns restricting a particular activity (or the process) (i.e., *Duration*, *Fixed Date Element*, and *Schedule Restricted Element*) may be configured with the corresponding *properties editor*. In Fig. 1, for example, activity `prepare treatment` is selected and the properties editor is shown in the lower part. In the upper section of this editor, the duration of the activity must be specified. In ATAPIS, *durations* are specified through three values [7], which can be interpreted as follows: usually, activity durations are contingent, i.e., it is possible to set up a duration range for any activity, the *contingent minimum* and *maximum duration*, however, the PAIS becomes aware of the effective activity duration only after its completion. Hence, the duration must not be restricted by the PAIS. In practice, however, activity durations usually represent worst case estimates; i.e., most durations may be restricted to some extend; we call this the *flexible maximum duration*. For better usability, the duration of the activity is visualized on the right side section of the properties editor.

In the lower section of the properties editor (cf. Fig. 1), two *fixed date elements* are specified for activity `perform treatment`. The one restricting the earliest start date of the activity retrieves its value from data element `date`. In turn, the *fixed date element* restricting the latest start date is set relatively to (i.e., 30 minutes after) the one on the earliest start date of the activity.

Similarly a *schedule restricted element* may be specified for any activity. In turn, this constraint is based on a language for representing collections of temporal intervals [9] (e.g., formula *[1-5]/days:during:weeks* represents the first 5 days of each week; i.e., Monday–Friday).

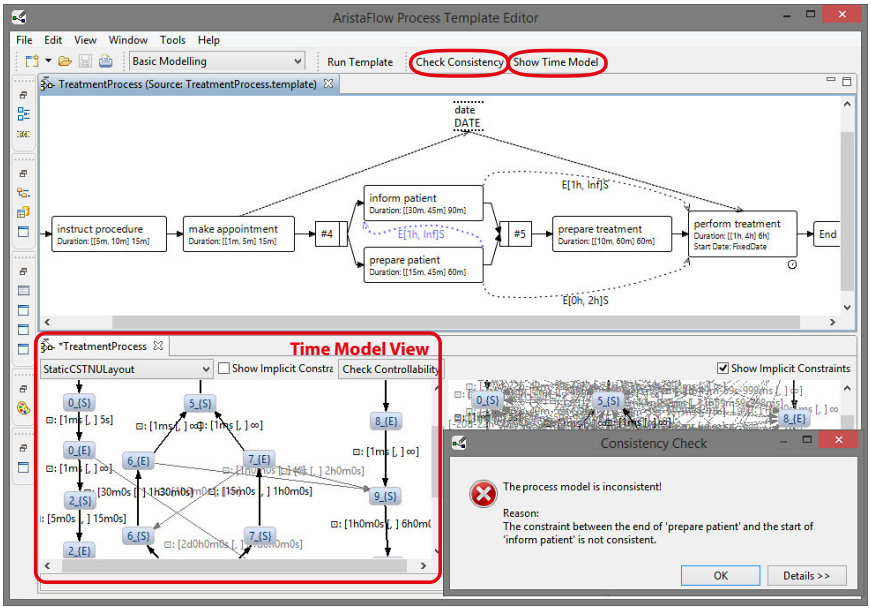

Fig. 2. Checking temporal consistency of a process schema

2.2 Checking Temporal Consistency

A time-aware process schema is enacted by performing its activities in the specified order, while obeying the specified temporal constraints. Generally, a time-aware process schema is denoted as *temporally consistent* if it is possible to perform all *execution paths* without violating the temporal constraints involved [1]. Note that temporal consistency of a time-aware process schema and its instances constitutes a fundamental prerequisite for a robust and error-free execution [1,5]. For any PAIS supporting time-aware processes, therefore, a crucial task is to check temporal consistency of the process schema at design time as well as to monitor and ensure temporal consistency of process instances during run time. This is challenging since temporal constraints might interact with each other resulting in complex (hidden) interdependencies. For example, assume that a time lag is added between activities `prepare patient` and `inform patient` in the process schema depicted in Fig. 1 (cf. Fig. 2). Assume further that the time lag specifies that activity `prepare patient` must be completed at least one hour before `inform patient` may be started. In this scenario, —although not directly obvious—the process schema can never be enacted without violating at least one of its constraints, i.e., the process schema is temporally inconsistent (cf. Fig. 2).

To check whether a particular process schema is temporally consistent, ATAPIS maps it to a specific *time model* (cf. Fig. 2). The latter allows us to capture the complex interdependencies between constraints, which are not explicit in process models. To support various consistency notions, the ATAPIS Toolset provides different implementations of the time model; e.g., using *conditional simple temporal networks* [11] to check for weak or dynamic consistency or *conditional simple*

temporal networks with uncertainty [3] to check for dynamic controllability—a more restricted form of temporal consistency. Particularly, we choose these models since they allow us to exploit and reuse correct and sound *checking algorithms* for well founded models representing temporal constraints.

When analyzing the temporal perspective of a process schema one may also view the corresponding time model including any hidden temporal constraints resulting from interdependencies of the specified temporal constraints (cf. Fig. 2). To support a thorough analysis of the temporal perspective of a process schema (e.g., to analyze the impact of a particular date for a *fixed date element*), we additionally provide editing capabilities for these time models.

3 Conclusion

The demonstration presents the ATAPIS Toolset and its components. It allows creating time-aware process schemas based on a well-founded modeling language. Further, it allows checking time-aware process schemas for temporal consistency in order to ensure their executability. We are currently investigating the impact of process change operations on time-aware processes and are implementing them in ATAPIS. Further, we are integrating temporal features with the run-time environment of the AristaFlow BPM Suite.

References

1. Bettini, C., Wang, X.S., Jajodia, S.: Temporal reasoning in workflow systems. Distrib Para Dat 11(3), 269–306 (2002)
2. Combi, C., Gozzi, M., Posenato, R., Pozzi, G.: Conceptual modeling of flexible temporal workflows. TAAS 7(2), 19:1–19:29 (2012)
3. Combi, C., Hunsberger, L., Posenato, R.: An algorithm for checking the dynamic controllability of a conditional simple temporal network with uncertainty. In: Proc ICAART'13 (2013)
4. Dadam, P., Reichert, M.: The ADEPT project: A decade of research and development for robust and flexible process support - challenges and achievements. Computer Science - R&D 22(2), 81–97 (2009)
5. Eder, J., Euthimios, P., Pozewaunig, H., Rabinovich, M.: Time management in workflow systems. In: Proc BIS'99. pp. 265–280 (1999)
6. Lanz, A., Kreher, U., Reichert, M., Dadam, P.: Enabling process support for advanced applications with the AristaFlow BPM Suite. In: BPM'10 Demo (2010)
7. Lanz, A., Posenato, R., Combi, C., Reichert, M.: Controllability of time-aware processes at run time. In: Proc CoopIS'13. pp. 39–56 (2013)
8. Lanz, A., Weber, B., Reichert, M.: Time patterns for process-aware information systems. Req Eng 19(2), 113–141 (2014)
9. Leban, B., McDonald, D., Forster, D.: A representation for collections of temporal intervals. In: Proc AAAI'86. pp. 367–371 (1986)
10. Reichert, M., Dadam, P.: ADEPTflex – supporting dynamic changes of workflows without losing control. J Intell Inf Syst 10(2), 93–129 (1998)
11. Tsamardinos, I., Vidal, T., Pollack, M.: CTP: A new constraint-based formalism for conditional, temporal planning. Constraints 8(4), 365–388 (2003)